This listing of claims will replace all prior versions, and listings, of claims in the application:

Listing of Claims:

1. (Currently Amended) A method for sharing resources on a multithreaded CPU capable of executing a plurality of threads, the method comprising:

deferring a yield comprising relinquishing use of the multithreaded CPU by ~~of~~ a first thread executing on the multithreaded CPU while waiting for at least a second thread executing on the multithreaded CPU to become ready to yield; and

yielding the first thread in response to at least the second thread becoming ready to yield.

2. (Original) The method according to claim 1, further comprising monitoring the plurality of threads for an occurrence.

3. (Original) The method according to claim 2, wherein the occurrence is a spin lock or an idle loop.

4. (Original) The method according to claim 2, further comprising making a yield call in response to the occurrence.

5. (Original) The method according to claim 1, further comprising marking storage of the first thread in response to receiving the yield call to indicate that the first thread is ready to yield.

6. (Original) The method according to claim 1, further comprising spinning the first thread while waiting for at least the second thread to become ready to yield.

7. (Original) The method according to claim 1, further comprising abandoning the yield call in response to detecting an event.

8. (Original) The method according to claim 7, wherein the event is a time-out or an external interrupt.

9. (Original) The method according to claim 7, further comprising returning control of the first thread to an operating system in response to detecting the event.

10. (Original) The method according to claim 9, further comprising saving the state of the operating system in response to detecting that at least the second thread is ready to yield.

11. (Original) The method according to claim 1, further comprising idling at least the first and second threads within a common virtual space in response to at least the second thread being ready to yield.

12. (Original) The method according to claim 11, further comprising idling all threads executing on the multithreaded CPU within the common virtual space.

13. (Currently Amended) A method for yielding a thread within a multithreaded CPU data processing system, wherein each of a plurality of threads executing on a multithreaded CPU must execute within a common virtual space, the method comprising:

deferring a yield comprising relinquishing use of the multithreaded CPU by ~~of~~ a thread while at least a subset of the plurality of threads yield; and

abandoning the yield of the thread in response to detecting an event while the yield is deferred.

14. (Original) The method according to claim 13, further comprising yielding the thread after the subset of threads yield, if the subset of threads yield prior to the event.

15. (Original) The method according to claim 13, wherein the event is selected from among a group consisting of: a time-out, an I/O interrupt and a combination thereof.

16. (Currently Amended) An apparatus comprising:

a computer having a multithreaded CPU, wherein the CPU is configured to execute a plurality of threads; and

a program resident in the computer, the program configured to defer a yield comprising relinquishing use of the multithreaded CPU by ~~of~~ a first thread of the plurality while waiting for at least a second thread of the plurality to become ready to yield; and further to initiate the yield of the first thread in response to at least the second thread of the plurality becoming ready to yield.

17. (Original) The apparatus according to claim 16, wherein the program initiates monitoring the plurality of threads for an occurrence.

18. (Original) The method according to claim 17, wherein the occurrence is a spin lock or an idle loop.

19. (Original) The apparatus according to claim 17, wherein the program initiates a yield call in response to the occurrence.

20. (Original) The apparatus according to claim 16, wherein the program initiates marking storage of the first thread in response to receiving the yield call to indicate that the first thread is ready to yield.

21. (Original) The apparatus according to claim 16, wherein the program initiates spinning the first thread while waiting for at least the second thread of the plurality to become ready to yield.

22. (Original) The apparatus according to claim 16, wherein the program initiates abandoning the yield call in response to detecting an event.

23. (Original) The apparatus according to claim 22, wherein the event is a time-out or an external interrupt.

24. (Original) The apparatus according to claim 22, wherein the program initiates returning control of the first thread to an operating system in response to detecting the event.

25. (Original) The apparatus according to claim 24, wherein the program initiates saving the state of the operating system in response to detecting that at least the second thread is ready to yield.

26. (Original) The apparatus according to claim 16, wherein the program initiates idling at least the first and second threads of the plurality within a common virtual space in response to at least the second thread of the plurality being ready to yield.

27. (Original) The apparatus according to claim 26, wherein the program initiates idling all threads of the plurality of threads within the common virtual space.

28. - 30. (Cancelled)

31. (Currently Amended) A program product, comprising:

(a) a program for yielding a thread within a multithreaded CPU data processing system, wherein each of a plurality of threads that execute on a multithreaded CPU must execute within a common virtual space, wherein the program is configured to defer a yield comprising relinquishing use of the multithreaded CPU by ~~of~~ a first thread of the plurality while waiting for at least a second thread of the plurality to become ready to yield; and further to initiate the

yield of the first thread in response to at least the second thread becoming ready to yield; and

(b) a tangible signal bearing medium bearing the first program.

32. (Cancelled)

33. (Previously Presented) The apparatus according to claim 16, wherein the program is configured to ensure that the plurality of threads execute within a common virtual space.